```
import pandas as p
import numpy as n
import matplotlib.pyplot as plt
import plotly.express as px
import seaborn as sns
```

```
d = p.read_excel("/content/sales1.xlsx")
```

```
#tail() which will give the last five records
d.tail()
```

	Row ID	Order ID	Order Date	Dispatch Date	Delivery Mode	Customer ID	Customer Name	Segment	City	State/Province	...	Region	Product ID	Cat
9995	9990	ES-2021-5488768	2021	2021	First Class	SN-20710	Steve Nguyen	Home Office	Northampton	England	...	North	OFF-BI-10001253	Su
9996	9991	ES-2021-5488768	2021	2021	First Class	SN-20710	Steve Nguyen	Home Office	Northampton	England	...	North	FUR-CH-10002373	Fu
9997	9992	ES-2021-4951531	2021	2021	Second Class	JF-15565	Jill Fjeld	Consumer	Preston	England	...	North	FUR-BO-10002003	Fu
9998	9993	ES-2021-2785118	2021	2021	Second Class	JH-16180	Justin Hirsh	Consumer	Castelnau-le-Lez	Languedoc-Roussillon-Midi-Pyrénées	...	Central	OFF-PA-10001661	Su
9999	9995	IT-2021-5726048	2021	2021	Second Class	DK-12985	Darren Koutras	Consumer	Dublin	Dublin	...	North	OFF-AR-10000715	Su

5 rows × 21 columns

```
#head() gives us the first five records
d.head()
```

	Row ID	Order ID	Order Date	Dispatch Date	Delivery Mode	Customer ID	Customer Name	Segment	City	State/Province	...	Region	Product ID	Categ
0	32	ES-2018-2801336	2018	2018	Standard Class	DB-13270	Deborah Brumfield	Home Office	Barcelona	Catalonia	...	South	OFF-BI-10002225	Of Supp
1	44	ES-2018-3186072	2018	2018	Standard Class	CA-12265	Christina Anderson	Consumer	Littlehampton	England	...	North	TEC-CO-10000620	Technol
2	45	ES-2018-3186072	2018	2018	Standard Class	CA-12265	Christina Anderson	Consumer	Littlehampton	England	...	North	TEC-AC-10004203	Technol
3	46	ES-2018-3186072	2018	2018	Standard Class	CA-12265	Christina Anderson	Consumer	Littlehampton	England	...	North	TEC-CO-10001596	Technol
4	47	ES-2018-5235241	2018	2018	Same Day	TB-21175	Thomas Boland	Corporate	Taverny	Ile-de-France	...	Central	OFF-SU-10004818	Of Supp

5 rows × 21 columns

- The above data set is about the sales of the Super-market store. Here we have 2 years of sales data like OrderID, Order Date, Dispatch Date, Delivery Mode, Customer ID, Customer Name, Segment City, State/Province, Region, Product ID, Category, Sub-Category, Product Name, Sales, Quantity, Discount, Profit, Returns.
- By using the above details of the Super-market store, we can analysis the trends and we can know the hidden patterns. By visualizing the data in the form of various charts which will give more information about the relation between the attributes, trends, patterns and we can explore about the sales, profit and other measures.

```
d.describe()
```

	Row ID	Order Date	Dispatch Date	Sales	Quantity	Discount	Profit
count	10000.00000	10000.000000	10000.00000	10000.000000	10000.000000	10000.000000	10000.000000
mean	5000.50000	2019.798000	2019.81480	293.808906	3.777300	0.103105	37.282974
std	2886.89568	1.094311	1.10219	486.055635	2.203268	0.174782	178.100384
min	1.00000	2018.000000	2018.00000	2.955000	1.000000	0.000000	-3059.820000
25%	2500.75000	2019.000000	2019.00000	49.462500	2.000000	0.000000	1.320000
50%	5000.50000	2020.000000	2020.00000	119.355000	3.000000	0.000000	14.220000
75%	7500.25000	2021.000000	2021.00000	320.708625	5.000000	0.100000	48.510000
max	10000.00000	2021.000000	2022.00000	7958.580000	14.000000	0.850000	3979.080000

```
d.count()
```

```
Row ID            10000
Order ID          10000
Order Date        10000
Dispatch Date     10000
Delivery Mode     10000
Customer ID       10000
Customer Name     10000
Segment           10000
City              10000
State/Province    10000
Country/Region    10000
Region            10000
Product ID        10000
Category          10000
Sub-Category      10000
Product Name      10000
Sales             10000
Quantity          10000
Discount          10000
Profit            10000
Returns           10000
dtype: int64
```

- The data set don't have any null values which will helps us to in analysising and visualizing the data easily. We can get accurate results.
- The next part is to convert the data. And need to explore the data for the further analysis.

```
print("Sum of the sales:",d["Sales"].sum())
print("Average of the sales:",d["Sales"].mean())
print("Maximum of the sales:",d["Sales"].max())
print("Minimum of the sales:",d["Sales"].min())
```

```
Sum of the sales: 2938089.0615000003
Average of the sales: 293.80890615000004
Maximum of the sales: 7958.58
Minimum of the sales: 2.955
```

```
#No, of sales with respect to Order Date
d.groupby('Order Date').sum()['Sales'].reset_index()
```

```
<ipython-input-8-8627eec84e97>:2: FutureWarning: The default value of numeric_only in DataFrameGroupBy.sum is deprecated. In a futur
  d.groupby('Order Date').sum()['Sales'].reset_index()
```

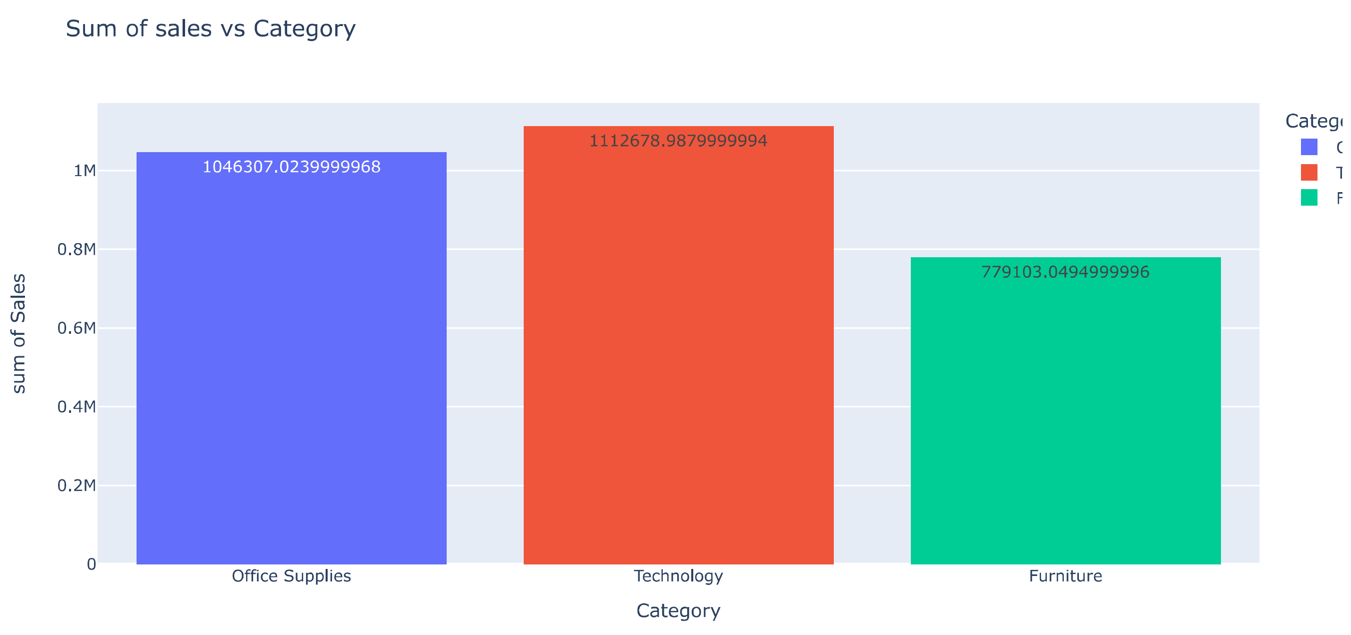
	Order Date	Sales
0	2018	4.777967e+05
1	2019	6.526474e+05
2	2020	7.654412e+05
3	2021	1.042204e+06

```
#No, of sales with respect to Order Date
d.groupby('Order Date').mean()['Sales'].reset_index()
```

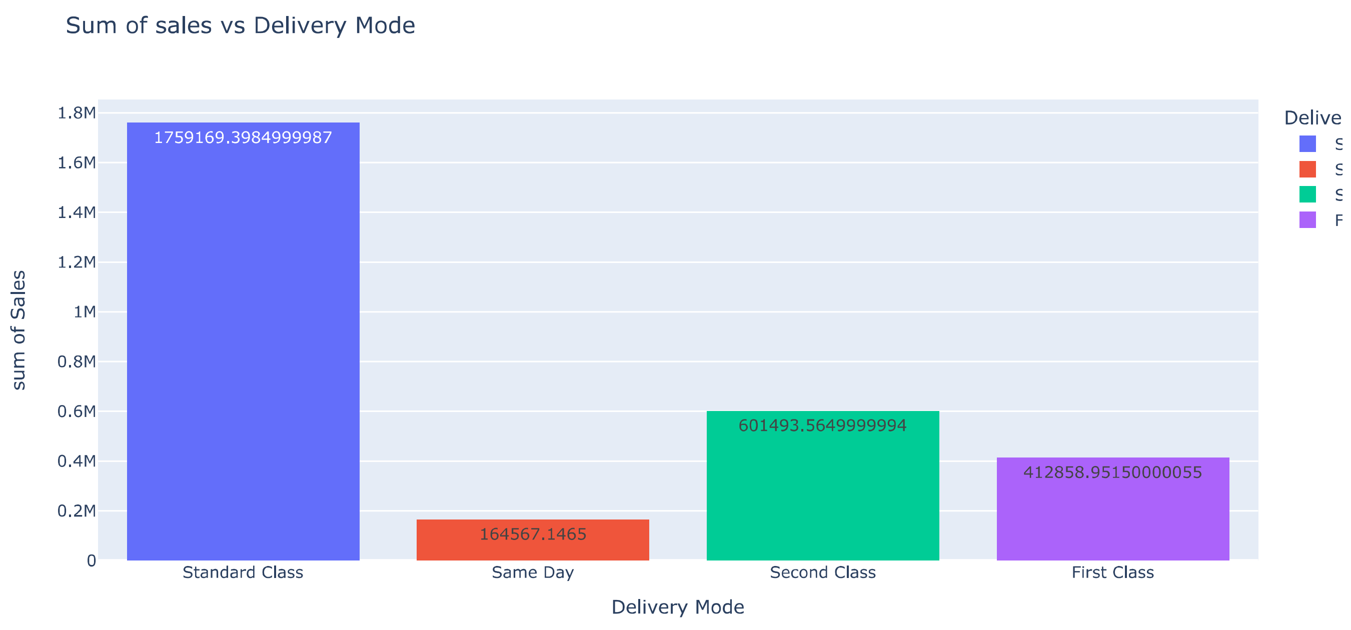
```
<ipython-input-9-f877dc722ed0>:2: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is deprecated. In a futu
  d.groupby('Order Date').mean()['Sales'].reset_index()
```

	Order Date	Sales
0	2018	289.048230
1	2019	291.100520
2	2020	297.028034
3	2021	295.409220

```
#Sum of sales regarding to each category
px.histogram(d,x='Category',y='Sales',color='Category',title="Sum of sales vs Category",text_auto="Sales")
```



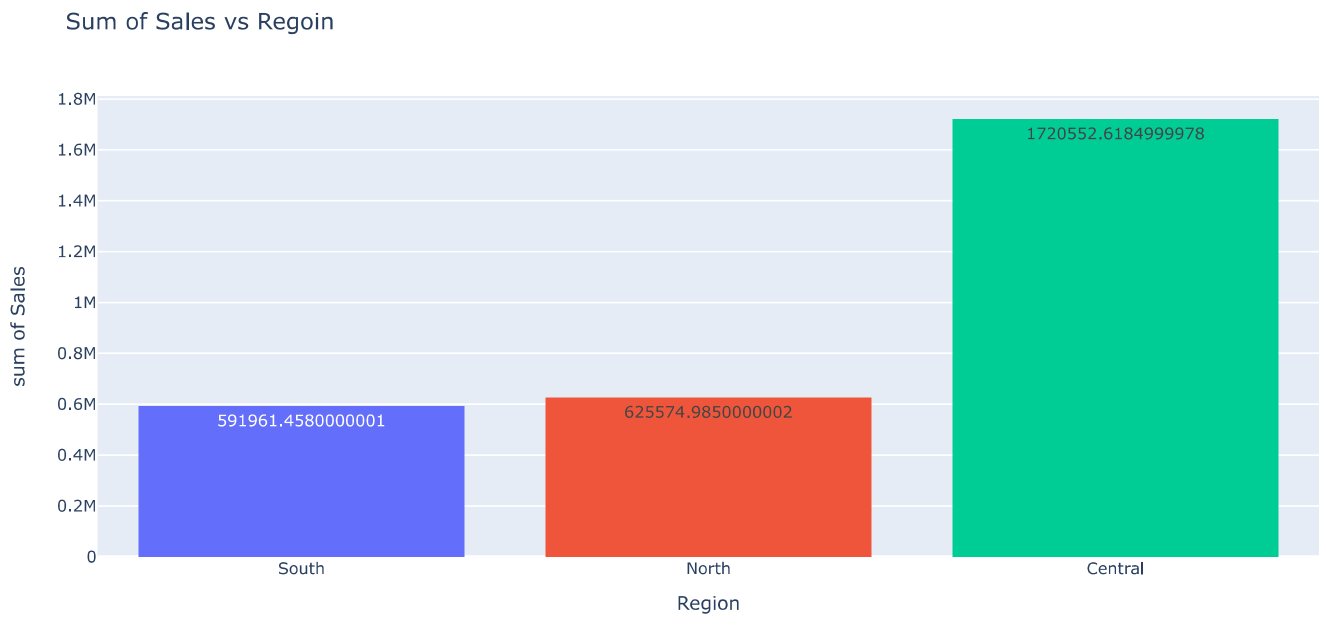
```
#Sum of sales regarding to each Delivery Mode
px.histogram(d,x='Delivery Mode',y='Sales',color='Delivery Mode',title="Sum of sales vs Delivery Mode",text_auto="Sales")
```



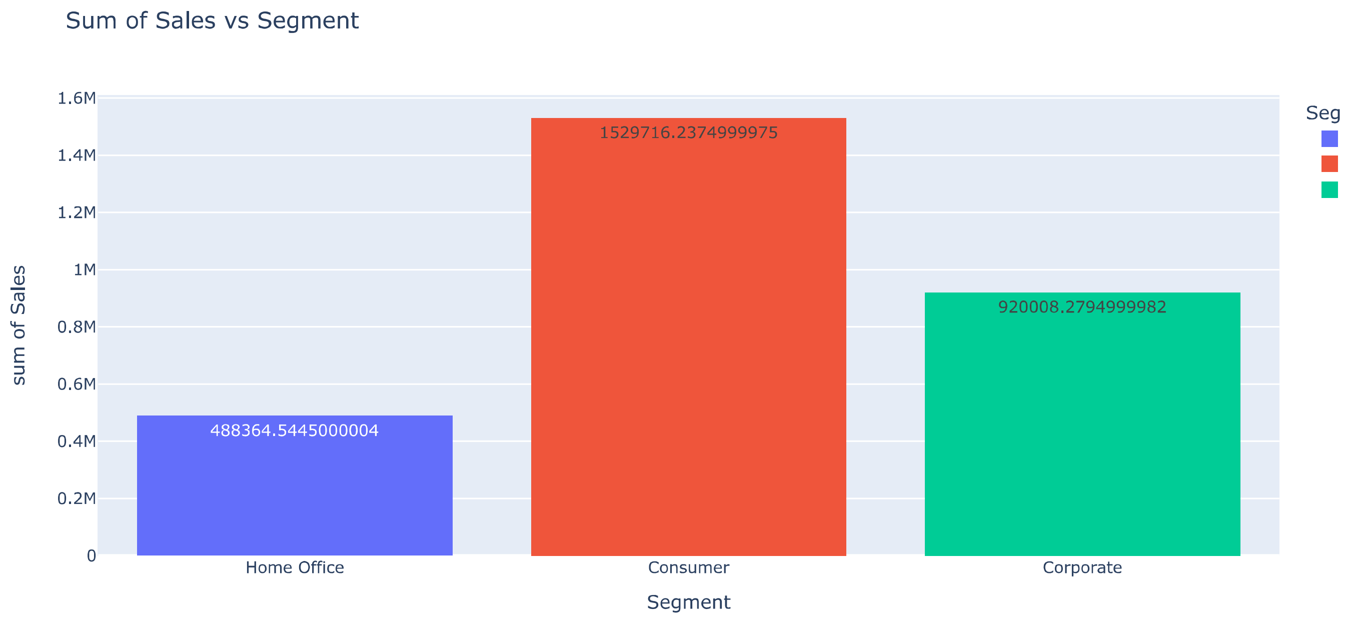
```
d.groupby(["Region"])["State/Province"].count().reset_index()
```

	Region	State/Province
0	Central	5822
1	North	2141
2	South	2037

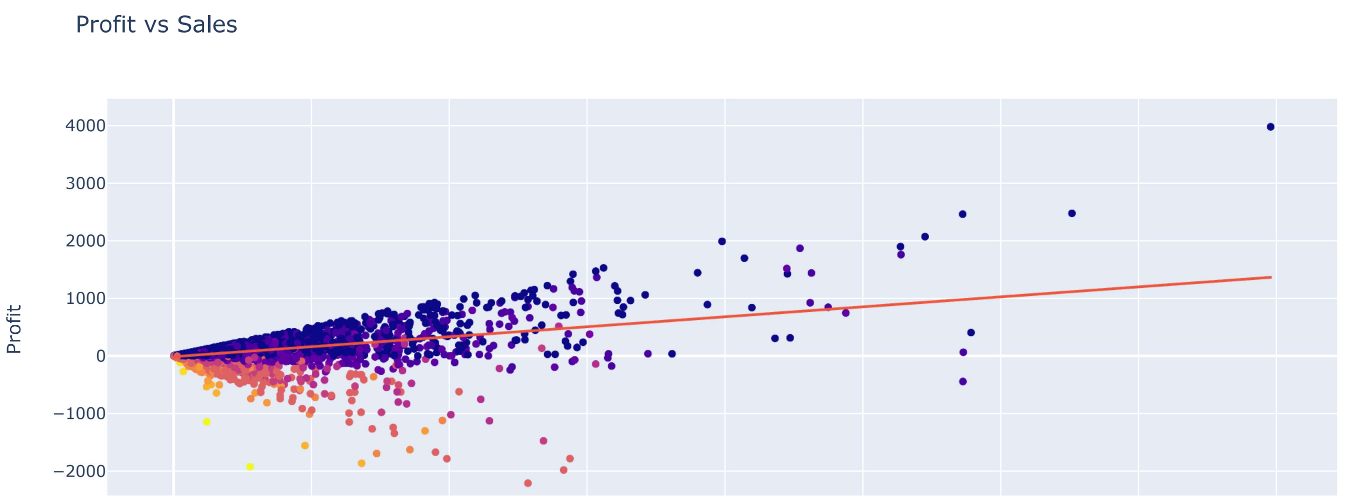
```
#Sum of sales regarding to each Region
px.histogram(d,x="Region",y= 'Sales',color='Region',title="Sum of Sales vs Regoin",text_auto="Sales")
```



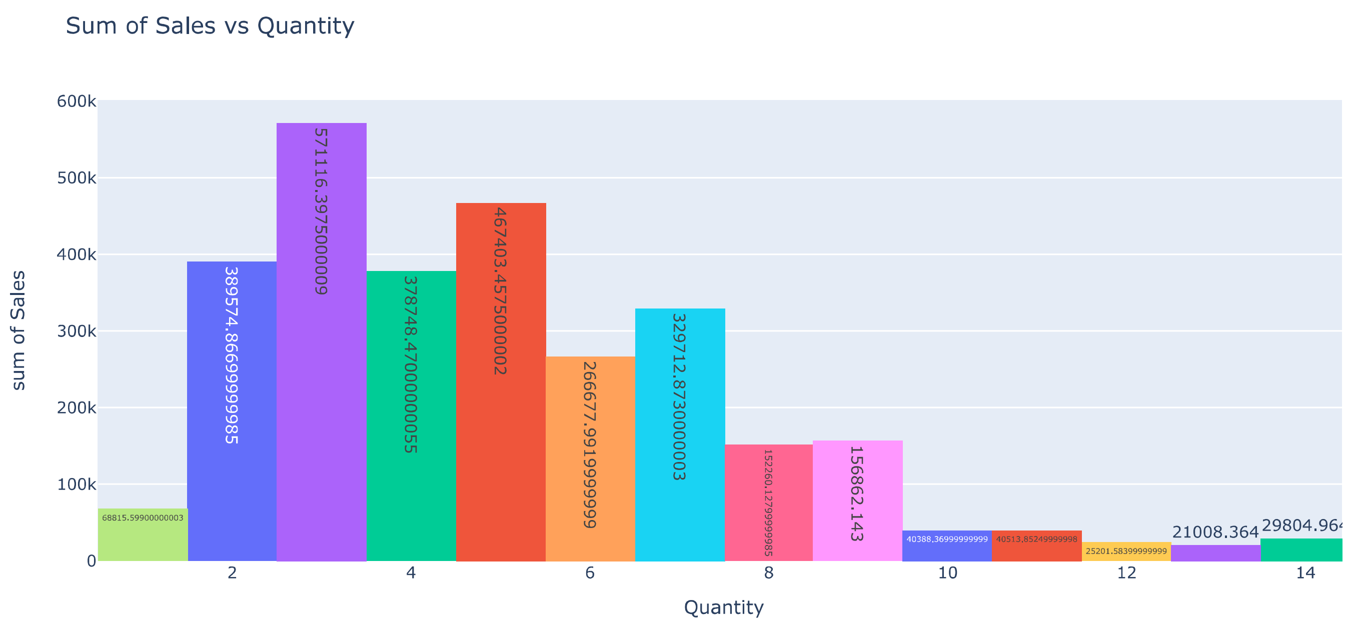
```
#Sum of sales regarding to each Segment
px.histogram(d, x="Segment", y="Sales",color='Segment',title="Sum of Sales vs Segment",text_auto="Sales")
```



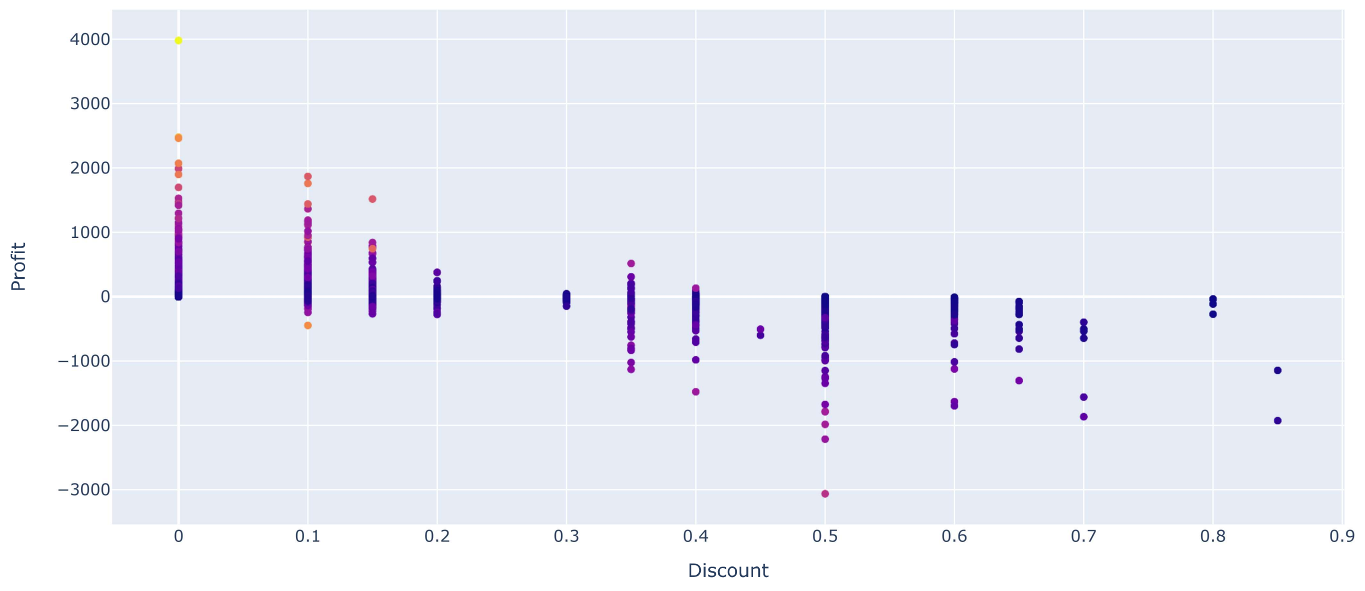
```
px.scatter(d,x="Sales",y="Profit",color='Discount',trendline="ols",title="Profit vs Sales")
```



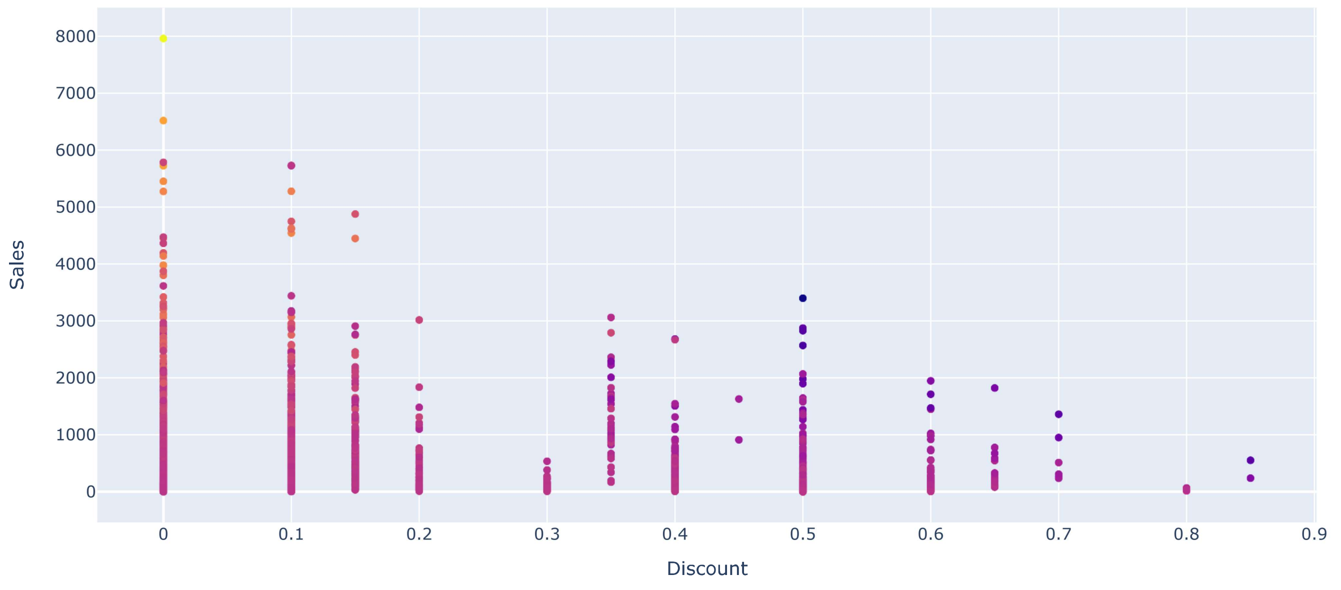
```
px.histogram(d,y="Sales",x="Quantity",color="Quantity",title="Sum of Sales vs Quantity",text_auto="Sales")
```



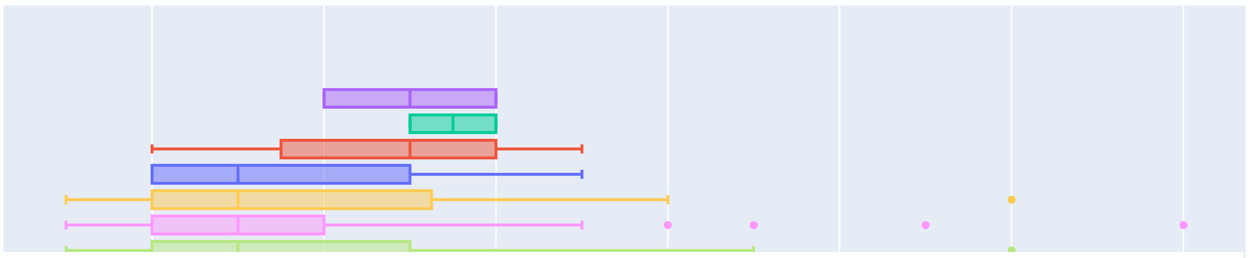
```
px.histogram(d,y="Profit",x="Quantity",color="Quantity",title="Sum of Sales vs Quantity",text_auto="Profit")
px.scatter(d,x="Discount",y="Profit",color='Sales')
```



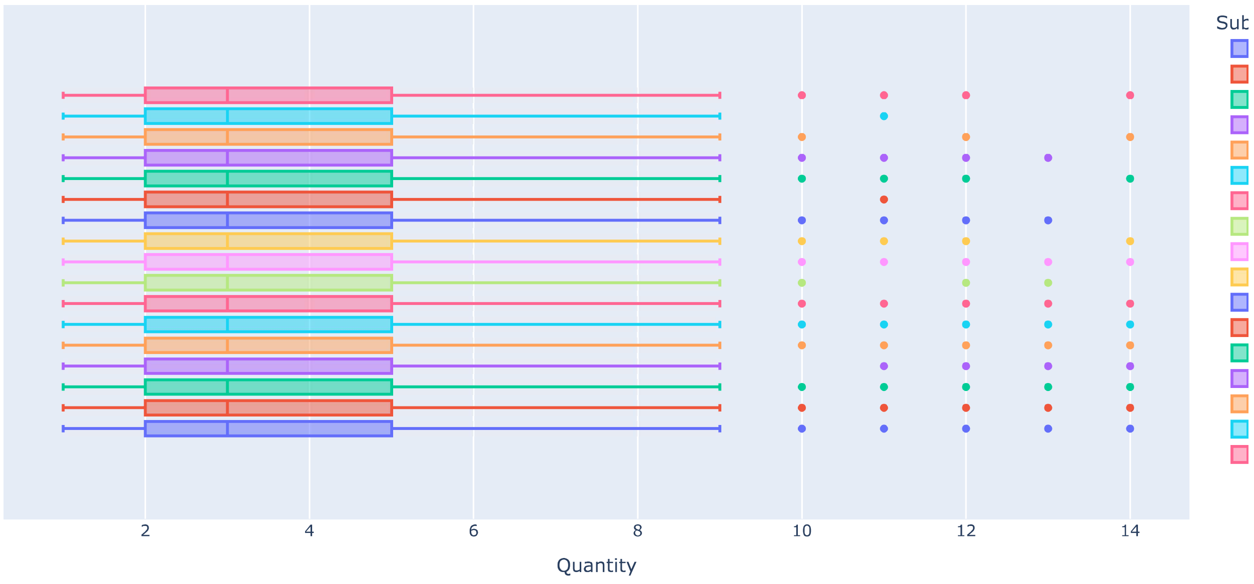
```
px.scatter(d,x="Discount",y="Sales",color='Profit')
```



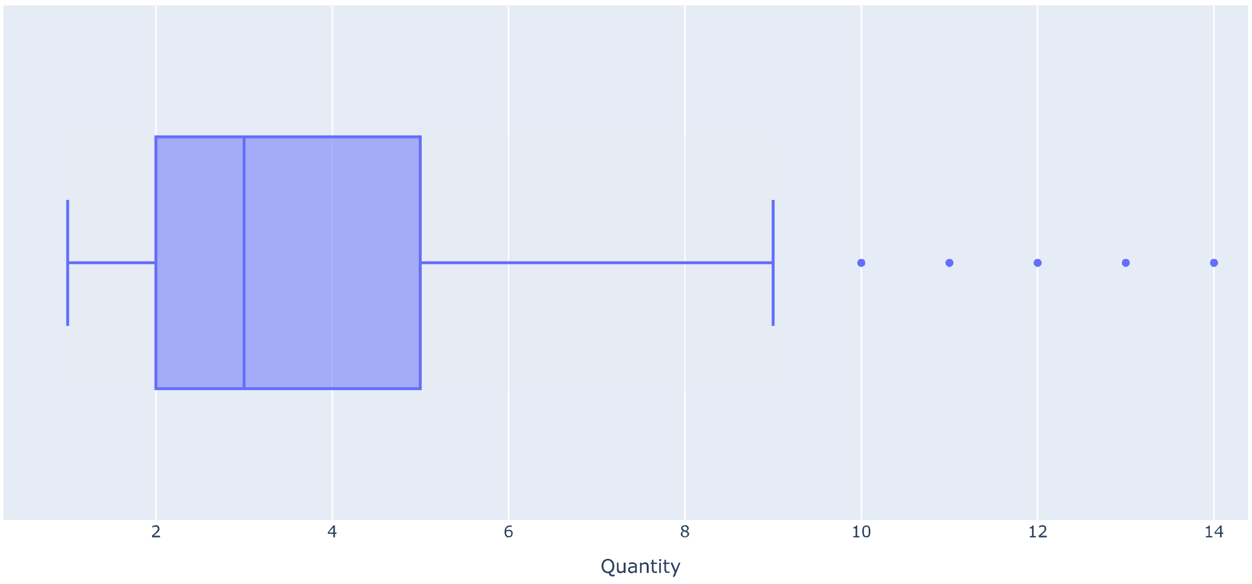
```
fig = px.box(d, x="Quantity",color="Discount")
fig.show()
```



```
fig = px.box(d, x="Quantity",color="Sub-Category")
fig.show()
```



```
fig = px.box(d, x="Quantity")
fig.show()
```



```
C_Sales = d.groupby(["Order Date","Category"]).sum()["Sales"].reset_index()
```

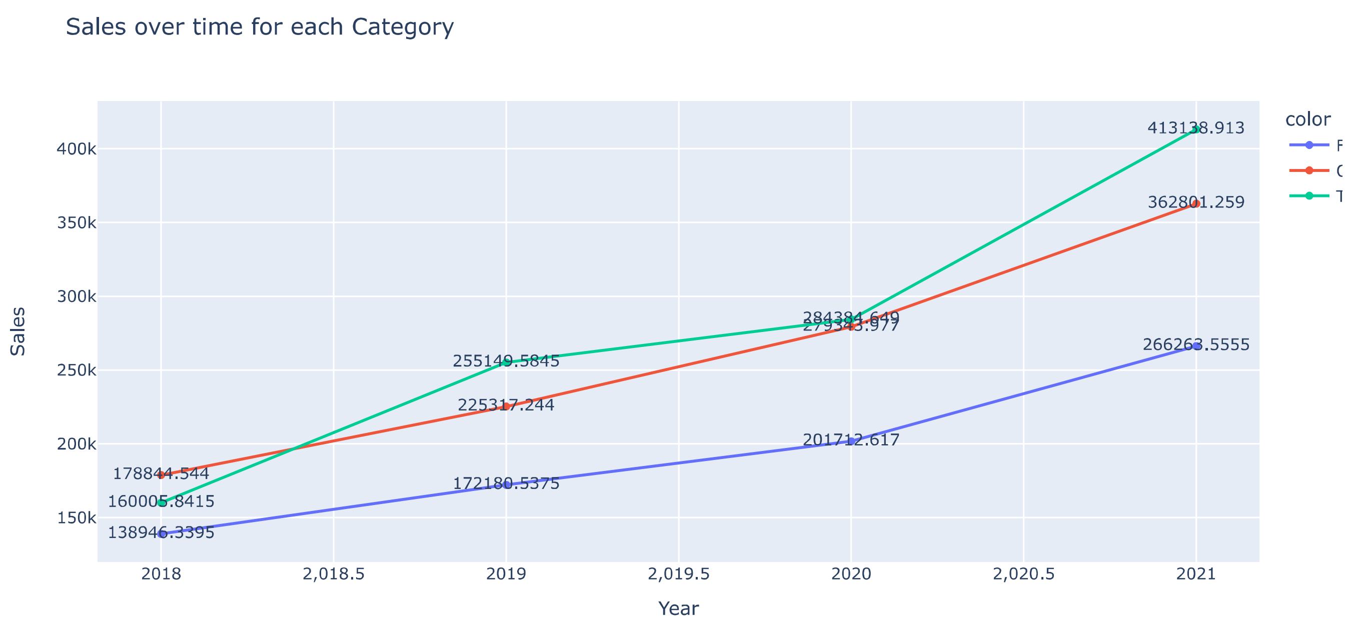
```
    <ipython-input-23-2e786e30873c>:1: FutureWarning:
```

```
The default value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, numeric_only will default to False. E
```

```
C_Sales
```

	Order Date	Category	Sales
0	2018	Furniture	138946.3395
1	2018	Office Supplies	178844.5440
2	2018	Technology	160005.8415
3	2019	Furniture	172180.5375
4	2019	Office Supplies	225317.2440
5	2019	Technology	255149.5845
6	2020	Furniture	201712.6170
7	2020	Office Supplies	279343.9770
8	2020	Technology	284384.6490
9	2021	Furniture	266263.5555
10	2021	Office Supplies	362801.2590
11	2021	Technology	413138.9130

```
px.line(d,x=C_Sales["Order Date"],y=C_Sales["Sales"],color=C_Sales["Category"],labels={"x":"Year","y":"Sales"},text=C_Sales["Sales"],tit
```



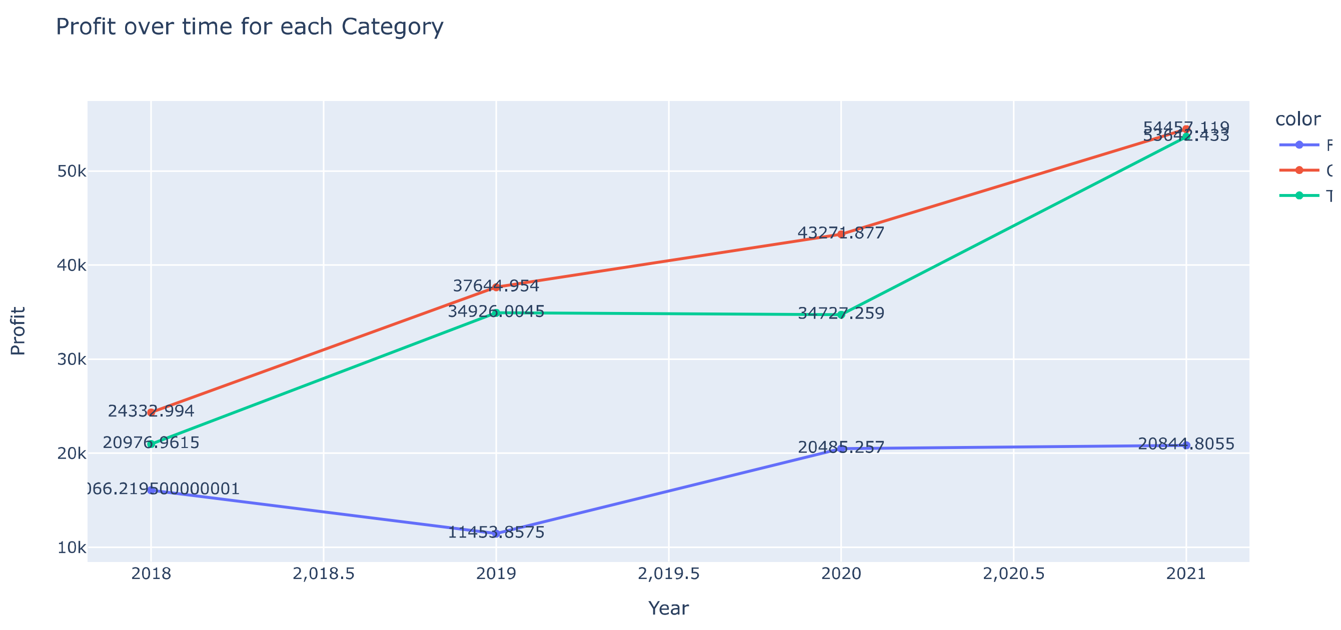
```
C_Profit = d.groupby(["Order Date","Category"]).sum()["Profit"].reset_index()
```

```
<ipython-input-26-c066a28dc280>:1: FutureWarning:

The default value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, numeric_only will default to False. E
```

```
C_Profit
```

	Order Date	Category	Profit
0	2018	Furniture	16066.2195
1	2018	Office Supplies	24332.9940
2	2018	Technology	20976.9615
3	2019	Furniture	11453.8575
4	2019	Office Supplies	37644.9540
5	2019	Technology	34926.0045

```
px.line(d,x=C_Profit["Order Date"],y=C_Profit["Profit"],color=C_Profit["Category"],labels={"x":"Year","y":"Profit"},text=C_Profit["Profi
```



```
Sub_Sales = d.groupby(["Order Date","Sub-Category","Category"]).sum()["Sales"].reset_index()
```

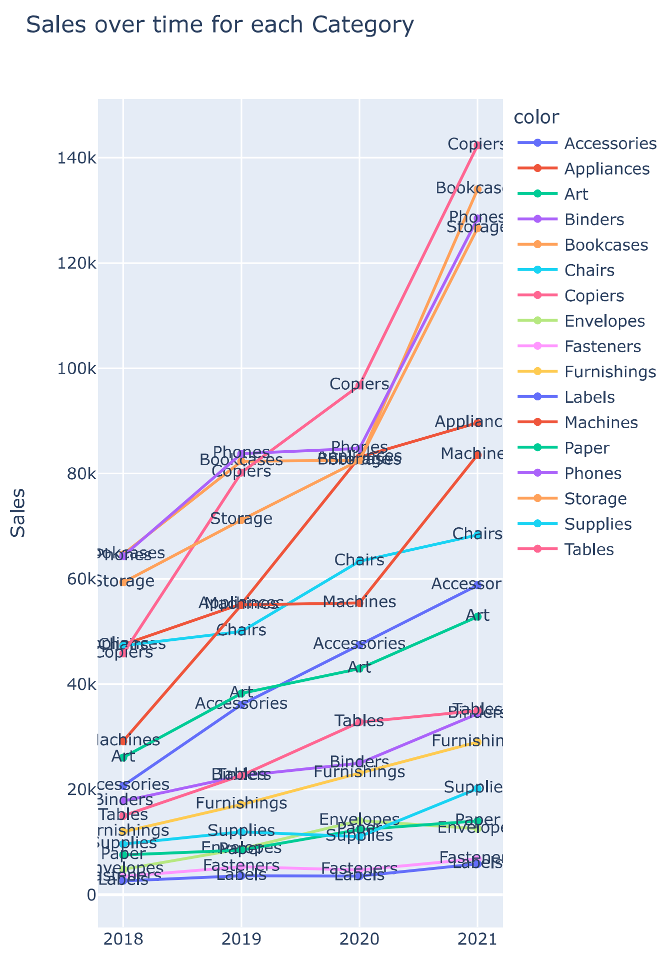
```
<ipython-input-29-3df56c31326c>:1: FutureWarning:

The default value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, numeric_only will default to False. E
```

```
Sub_Sales
```

	Order Date	Sub-Category	Category	Sales
0	2018	Accessories	Technology	20704.1040
1	2018	Appliances	Office Supplies	47483.2020
2	2018	Art	Office Supplies	26063.7480
3	2018	Binders	Office Supplies	17808.2430
4	2018	Bookcases	Furniture	64680.1590
...	...	...	...	...
63	2021	Paper	Office Supplies	13991.9010
64	2021	Phones	Technology	128460.7470
65	2021	Storage	Office Supplies	126608.6340
66	2021	Supplies	Office Supplies	20176.8810
67	2021	Tables	Furniture	34972.6125

68 rows × 4 columns

```
px.line(d,x=Sub_Sales["Order Date"],y=Sub_Sales["Sales"],color=Sub_Sales["Sub-Category"],labels={"x":"Year","y":"Sales"},text=Sub_Sales['
```



```
Sub_Profit = d.groupby(["Order Date","Sub-Category","Category"]).sum()["Profit"].reset_index()
```

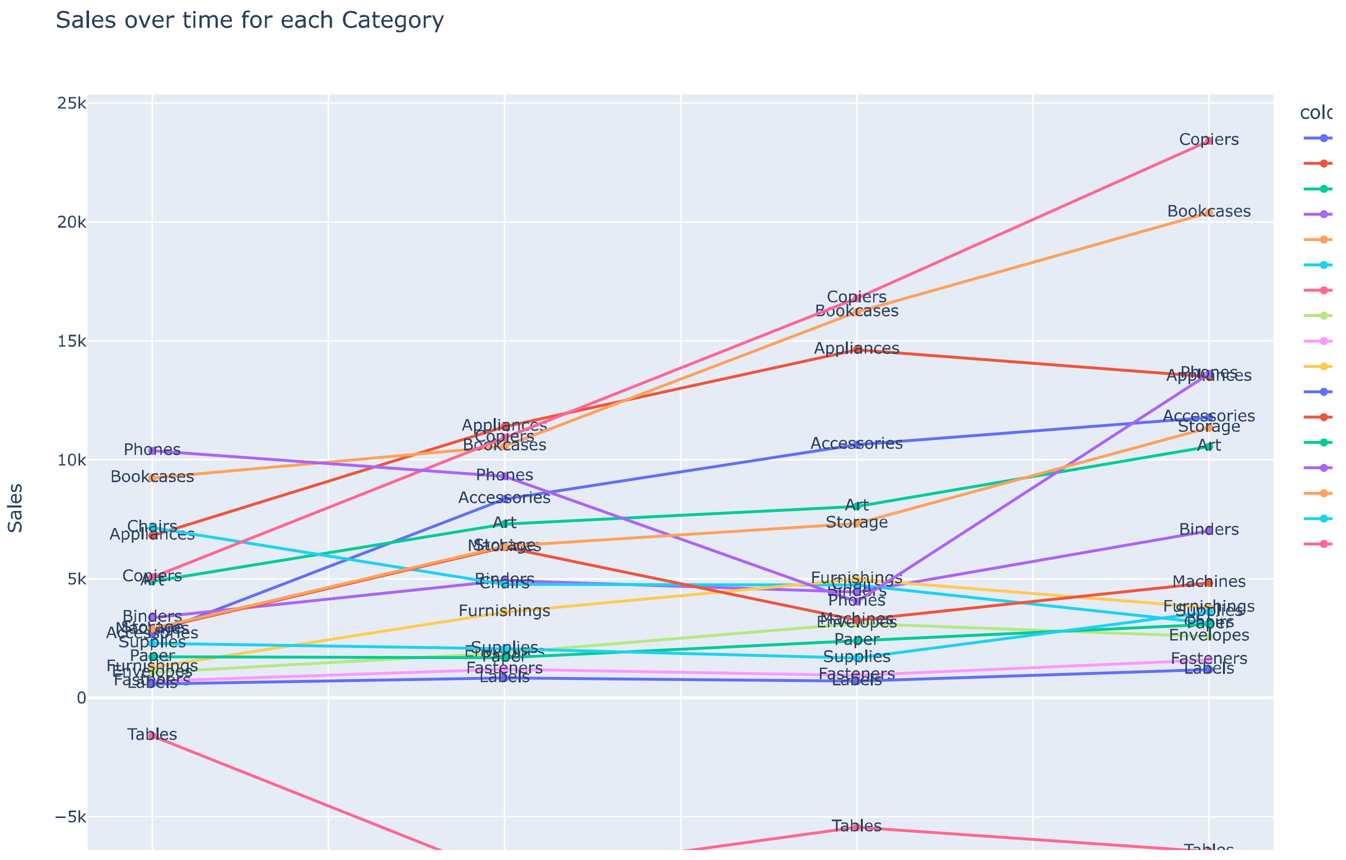
```
<ipython-input-32-32e8cc66f97d>:1: FutureWarning:

The default value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, numeric_only will default to False. E
```

```
Sub_Profit
```

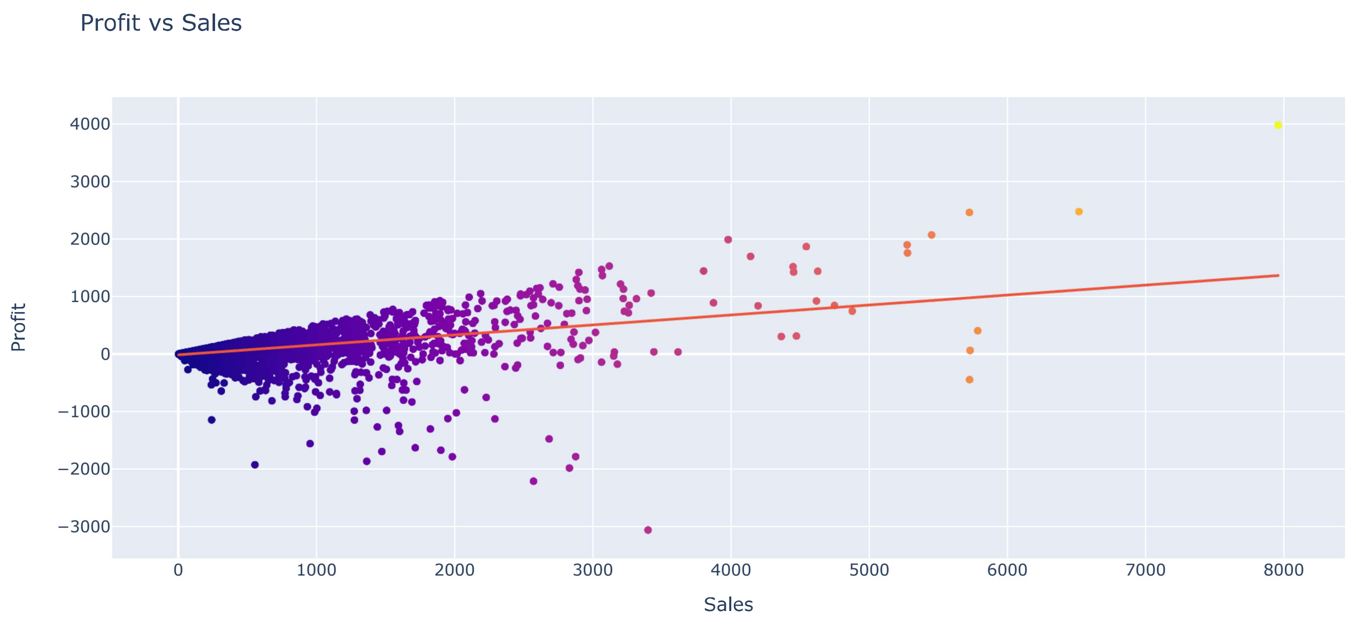
	Order Date	Sub-Category	Category	Profit
0	2018	Accessories	Technology	2667.5940
1	2018	Appliances	Office Supplies	6819.5220
2	2018	Art	Office Supplies	4893.4380
3	2018	Binders	Office Supplies	3364.8330
4	2018	Bookcases	Furniture	9236.7090
...	...	...	...	...
63	2021	Paper	Office Supplies	3086.7810
64	2021	Phones	Technology	13617.7770
65	2021	Storage	Office Supplies	11350.0140
66	2021	Supplies	Office Supplies	3609.0810
67	2021	Tables	Furniture	-6467.0475

68 rows × 4 columns

```
px.line(d,x=Sub_Profit["Order Date"],y=Sub_Profit["Profit"],color=Sub_Profit["Sub-Category"],labels={"x":"Year","y":"Sales"},text=Sub_Pro
```



```
px.scatter(d,x="Sales",y="Profit",color='Sales',trendline="ols",title="Profit vs Sales")
```


Profit vs Sales
Profit
Sales